NAME:________________________
Student ID:__________

This exam is closed book, notes, computer and cell phone.
Show partial solutions to get partial credit.
Make sure you have answered all parts of a question.
If your solutions are not written legibly, you won’t get full credit.
Clarity and succinctness will be rewarded.

Question 1:__________ (out of 20)
Question 2:__________ (out of 10)
Question 3:__________ (out of 15)
Question 4:__________ (out of 15)
Question 5:__________ (out of 10)
Question 6:__________ (out of 10)
Question 7:__________ (out of 10)
Question 8:__________ (out of 10)
Total: (out of 100)
1. Short questions:

(a) Give 4 operations efficiently implementable with the array implementation of a heap. For each operation give the cost in $O$-notation as a function of the size $n$ of the heap.

(b) What is the comparison tree lower bound for sorting and what is the key operation the sorting algorithm is allowed to use for determining the order of keys?

(c) Which of the following sorting algorithms are stable: InsertionSort, MergeSort, QuickSort, CountingSort, HeapSort?

(d) What is the worst-case and average-case running time of QuickSort? When does the worst-case occur?

(e) Give an algorithm for finding the largest and the second largest key among distinct $n$ keys? The straightforward algorithm requires $2n - 3$ comparisons. Give a 3 sentence description of an algorithm that requires fewer (how many?) comparisons.

Hint: Find largest using a tournament and ask yourself how many more comparisons it takes to find 2nd largests.
2. Show by induction that the number of leaves of a ternary tree of height $h$ is at most $3^h$.

Definition:
- In a *ternary tree* a node has at most 3 children.
- The *height of a tree* is the longest number of edges on any path from the root to a leaf.
3. Describe and algorithm that does the following. It is given \( n \) integers in the range 0 to \( k \). After preprocessing its input the algorithm must be able to answer any query of the form:

Input: Integers \( a, b \), such that \( 0 \leq a, b \leq k \).

Question: How many of the \( n \) integers fall into a range \([a..b]\).

Your algorithm is allowed to use \( O(n + k) \) preprocessing time and after this, each query must cost \( O(1) \) time.

Hint: The setup of the problem is similar to the one-dimensional interval summing problem covered in class. For the solution, you might use part of CountSort.

(a) Give your preprocessing procedure in pseudo-code and reason that it requires \( O(n + k) \) time.

(b) Show how to process the range queries in \( O(1) \) time.
4. Consider the following sorting algorithm for sorting \( A(1..n) \):

\[
\text{Bubblesort}(A, n)
\]
1. for \( i=1 \) to \( n-1 \)
2. for \( j=n \) downto \( i+1 \)
3. if \( A(j) < A(j-1) \)
4. exchange \( A(j) \) with \( A(j-1) \)

You are to prove the correctness of the algorithm using a loop invariant. Start by seeing what the algorithm does on some small examples. For the sake of simplicity assume the keys are distinct.

(a) The outer loop varies \( i \). What is the invariant (call it \( P(i) \)) that holds at the beginning of this loop?

(b) Complete the inductive argument by showing the Initialization, Maintenance and Termination part of the correctness proof.

(c) What is the running time of this algorithm in \( \Theta \) notation and why?

Note that this algorithm has the same time bound in \( \Theta \) notation no matter whether you consider worst-case or average case time (the same set of comparisons are always used for each execution).
5. Show how to sort the following array of numbers using RadixSort:

(a) Give the status of the array after each of the three passes:
   428
   525
   841
   315
   152
   582
   421
   141

(b) What is the running time of the algorithm if you are to sort $n$ $d$-digit numbers in the range from 0 to $k$. 
6. Show that $\sum_{i=1}^{n} \log_2(i + 2) = \Theta(n \log n)$. 
7. Show using the Substitution Method that $T(n) = O(n^2)$, where

$$T(n) = \begin{cases} 
1 & \text{for } n = 1 \\
4T\left(\left\lfloor \frac{n}{2} \right\rfloor \right) + n & \text{otherwise.}
\end{cases}$$

What is your exact conjecture? Don’t forget the base case!
For the sake of simplicity you can assume $n$ is a power of 2, i.e. $\left\lfloor \frac{n}{2} \right\rfloor = \frac{n}{2}$.
Hint: First try to prove the conjecture $T(n) \leq cn^2$, and then try to refine the conjecture.

8. You are to show how BuildHeap works on the following 10 keys:

\[3\ 8\ 9\ -1\ 5\ 6\ 10\ 1\ 0\ 9\]

Draw the following binary trees
(a) for the initial array
(b) after nodes 5 and 4 have been processed
(c) after 3 and 2 have been processed
(d) the final heap
(e) How expensive is the BuildHeap operation as a function of $n$?
(f) How expensive is BuildHeap if you inserted the $n$ keys one at a time?